04-13-00

A

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

jc525 U.S. PTO
09/547539
04/12/00

jc658 U.S. PTO
04/12/00

Applicant: SMITH ET AL.

Docket: 144243.1/40062.53US01

Title: METHOD AND SYSTEM FOR ACCEPTING PRECOMPILED INFORMATION

CERTIFICATE UNDER 37 CFR 1.10
'Express Mail' mailing label number: EL435543224US
Date of Deposit: April 12, 2000
I hereby certify that this paper or fee is being deposited with the United States Postal Service 'Express Mail Post Office To Addressee' service under 37 CFR 1.10 and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

By: Linda McCormick
Name: Linda McCormick

BOX PATENT APPLICATION
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

We are transmitting herewith the attached:

- ☒ Transmittal sheet, in duplicate, containing Certificate under 37 CFR 1.10.
- ☒ Utility Patent Application: Spec. 18 pgs; 68 claims; Abstract 1 pgs.
  The fee has been calculated as shown below in the 'Claims as Filed' table.
- ☒ 4 sheets of informal drawings
- ☒ An unsigned Combined Declaration and Power of Attorney
- ☒ A check in the amount of $1788.00 to cover the Filing Fee
- ☒ Return postcard

CLAIMS AS FILED

Number of Claims Filed		In Excess of:		Number Extra		Rate		Fee
Basic Filing Fee								$690.00
Total Claims								
68	–	20	=	48	x	18.00	=	$864.00
Independent Claims								
6	–	3	=	3	x	78.00	=	$234.00
MULTIPLE DEPENDENT CLAIM FEE								$0.00
TOTAL FILING FEE								$1788.00

Please charge any additional fees or credit overpayment to Deposit Account No. 13–2725. A duplicate of this sheet is enclosed.

MERCHANT & GOULD P.C.
P.O. Box 2903, Minneapolis, MN 55402–0903
(612) 332–5300

By: _____
Name: Erik G. Swenson
Reg. No.: 45,147
Initials: EGS:PSTkaw

MS No. 144243.1 / M&G No. 40062.53US01

METHOD AND SYSTEM FOR ACCEPTING PRECOMPILED INFORMATION

Technical Field

The present invention relates to distributed computing environments and more particularly to sharing processes in distributed computing environments.

Background

Typically, computing systems include a processor and a memory system. In some computing systems, the processor and memory system is relatively small and thus limits the capacity of the computing system to perform large computer processes, such as executing large computer programs or compiling a computer program. These computing systems are primarily intended to execute limited programs associated with their functionality. For example, a cellular phone executes a computer program associated with transmitting and receiving telephone calls and managing associated data, i.e. names and telephone numbers.

Some processes on computing systems are processor and memory intensive, often exceeding the capacity of the computing system or taking a considerable amount of time to complete the process. Therefore, improvements are desirable.

Summary

The compilation process is processor and memory intensive. This process consumes valuable resources of the computing subsystem, especially for small

devices. Small computing devices, such as a cellular telephone, might not have the processor, memory capacity, or power capacity to compile code. Therefore, the present invention contemplates methods and systems for offloading the compilation process.

In one aspect of the present invention, a method of offloading compilation is provided. The method includes transmitting compilation information from a first subsystem to a second subsystem. The method also includes compiling computer program code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem. The method further includes receiving the machine-executable code from the second subsystem into the first subsystem.

In another aspect, a computer program storage medium readable by a computing system and encoding a computer program of instructions for offloading compilation is provided. The computer process is analogous to the method described above.

Another aspect of the present invention includes a system for offloading compilation. The system includes a transmit module, a compile module, and a receive module. The transmit module transmits compilation information from a first subsystem to a second subsystem. The compile module compiles program code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem. The receive module receives the machine-executable code from the second subsystem into the first subsystem.

In another aspect, a computer data signal embodied in a carrier wave readable by a computing system and encoding a computer program of instructions for

offloading compilation is provided. The computer process is analogous to the method described above.

A more complete appreciation of the present invention and its scope may be obtained from the accompanying diagrams, that are briefly described below, from the following detailed descriptions of presently preferred embodiments of the invention and from the appended claims.

Brief Description of the Drawings

Fig. 1 is a schematic representation of a distributed computing environment that may be used to implement aspects of the present invention;

Fig. 2 is a schematic representation of a computing system that may be used to implement aspects of the present invention;

Fig. 3 is a schematic representation of methods and systems for offloading compilation, according to an embodiment of the present invention; and

Fig. 4 is a flow chart illustrating the logical operations of the methods and systems of Fig. 3.

Detailed Description

In the following description of preferred embodiments of the present invention, reference is made to the accompanying drawings that form a part hereof, and in which is shown by way of illustration specific embodiments in which the invention may be practiced. It is understood that other embodiments may be utilized and structural changes may be made without departing from the scope of the present

invention.

In general, the present disclosure describes methods and systems for accepting precompiled information from other systems, or in other words, offloading compilation from one subsystem to another subsystem.

Typically, computing systems execute a sequence of logical operations, commonly referred to as a computer program initially written in source language code. The source code could be any number of common computer programming languages, such as Fortran, C++, Pascal, Java, and the like. Each language has its own advantages and disadvantages; therefore, the particular source code language is a matter of choice.

The computing system is able to execute the sequence of logical operations from a machine-readable, or machine-executable, code compiled from the source code. By the term "machine-readable," it is meant that the code is in a format with which the computing system can identify and execute the sequence of instructions. There are a large variety of machine-readable formats. The machine-readable format required by any particular computing system, in order to execute, is dependent on the particular computing system that will execute the code. Code that is executable on one particular computing system might not be executable on another computing system.

Because each computing system requires its own particular machine-readable format and because there are many source language codes, the source language codes are typically translated by a source-code compiler into an intermediate language (IL), typically implemented by and accepted by a large variety of systems.

Typically, the IL code is distributed to any number of receiving systems. Again, because each computing system requires its own particular machine-readable

format, the IL code is translated by a machine-executable compiler. The machine-executable compiler translates the IL code into a format readable by the particular system desiring to execute the code.

In this fashion, each source code programmer needs to only have the particular source-code compiler for the particular source code desired to be translated into the IL code, rather than many source-code compilers for each type of machine-executable code. Likewise, each computing subsystem needs to only have the machine-executable compiler associated with the particular computing subsystem, rather than many machine-executable compilers for each type of source code.

It is noted that the machine-executable compiler does not necessarily compile the entire IL code into a machine-executable code prior to execution of the program. The machine-executable compiler might only compile the IL code as it is needed. For example, as the IL code is executed, the machine-executable compiler compiles the IL code into the machine-executable code. The machine-executable compiler compiles only the IL code portion being executed. Any unexecuted portions, such as subroutines, remain in IL format. This process has advantages and disadvantages. For example, this process saves overall compilation time, because the entire IL code is not compiled, but slows done execution of the program, because the compilation occurs during execution. This process also saves secondary storage space on the computing device, because the secondary device does not need to store the entire machine-executable code. In general, there is a trade-off between the quality of the compiled code produced and the amount of time taken for the compilation. Thus,

higher quality, or optimized, code results in longer compilation times and more memory resources required.

Referring now to Fig. 1, a distributed computing environment 200 might include a number of computing subsystems. In the particular embodiment illustrated in Fig. 1, the distributed computing environment 200 includes a first computing subsystem 201, a second computing subsystem 202, a third computing subsystem 203, and a fourth computing subsystem 204. Of course, any combination or any part of the subsystems 201-204 might be components of a single subsystem. Each computing subsystem 201-204 might be one of a variety of computing systems. Computing systems of this sort can include a server, a router, a network personal computer, a peer device, a hand-held device, a small device, and the like, or other common network nodes, each typically including many or all of the elements described herein in connection with the computing system 100 of Fig. 2, described hereinafter.

The distributed computing environment 200 might be one of a variety of distributed computing environments, for example, a local area network (LAN) 206 or a wide are network (WAN) 208. In such an environment the computing systems 201-204 are linked to each other using logical connections 211. Such computing environments are commonplace in offices, enterprise-wide computer networks, Intranets, and the Internet.

In the embodiment illustrated in Fig. 1, the first computing subsystem 201 and the second computing subsystem 202 are part of a LAN, while the first, second, and third computing subsystems 201-203 are part of a WAN. The fourth computing subsystem 204 might be a small device, occasionally connected to the first computing

subsystem 201.

Referring to Fig. 1, the first computing subsystem 201 includes a first compiler 221, capable of translating IL code into the machine-executable code readable by the first computing subsystem 201 and a second compiler 261, capable of translating Il code into machine-executable code readable by the fourth computing subsystem 204, as will be explained in more detail below. In one exemplary embodiment, the first and second compilers 221, 261 are one compiler that uses compilation information to compile code for different computing devices. In another embodiment, the third computing subsystem 203 includes a third compiler 223, capable of translating the IL code into the machine-executable code readable by the third computing subsystem 203.

It is noted that the fourth computing subsystem 204 does not include a compiler. The fourth computing subsystem 204 might be a small device, such as a cellular telephone, having limited processor and memory capacity. Preferably, the fourth computing subsystem 204 includes a compilation program 264, typically written in IL code. The compilation program 264 contains compilation instructions, or compilation information, for compiling, on another computing subsystem, i.e. the first computing subsystem 201, an IL code into machine-executable code for the fourth computing subsystem 204. The compilation program 264 might be written in IL code, of a higher level of source language.

In general, the fourth computing subsystem 204 uses a compiler from another subsystem having more processing and memory capacity, such as the second compiler 261 of the first computing subsystem 201, to offload compilation of IL code into machine-executable code for the fourth computing subsystem 204. The second

compiler 261 uses the compilation program 264 from the fourth computing subsystem 204, as will be explained in more detail below. By the term "offloading compilation," it is meant a subsystem, such as the fourth computing subsystem 204, has another subsystem, such as the first computing subsystem 201, compile code into machine-executable code for it.. The first subsystem 201 that performs the compilation is thus doing so either in parallel with network communication of the code, offsetting the performance/code compilation time tradeoff, or in isolation of the code's execution. This allows the first subsystem 201, having greater resources, to compile more optimal code for the fourth subsystem 204. In some instances, the first subsystem 201 might already have compiled the code. In such instances, the fourth subsystem 204 provides identification to the first subsystem 201 regarding the code the fourth subsystem 204 desires. The first subsystem 201 can transmit the compiled code to the fourth subsystem 204 without recompiling it.

Preferably, the fourth computing subsystem 204 also includes a receipt policy 244. The receipt policy 244 is a policy determined by the host computing subsystem, in this example, the fourth computing subsystem 204, by an administrator of the host computing subsystem, or by another subsystem, about which machine-executable code it will accept. Typically, the receipt policy 244 is based on the source of the receipt of the information or a trusted source. By the term "trusted source," it is meant the host fourth computing subsystem 204 has knowledge that machine-executable code received by the fourth computing subsystem 204, for example, code received from the first computing subsystem 201, will not compromise the integrity of the fourth computing subsystem 204. This knowledge may be based on a trusted relationship

between the computing subsystems, i.e. the first and fourth computing subsystems 201, 204.

For example, the transmitting and receiving computing subsystems may be connected via a secure link. Therefore, the receiving fourth computing subsystem 204 knows that the machine-executable code could not have been tampered with when received from the first computing subsystem 201. An exemplary policy might be that the receiving computing subsystem will accept machine-executable code from any subsystem with which it is securely linked.

Referring now to Fig. 2, an exemplary environment for implementing embodiments of the invention includes a general purpose computing device in the form of a computing system 100, such as a personal computer, includes at least one processing unit 102. A variety of processing units are available from a variety of manufacturers, for example, Intel or Advanced Micro Devices. The computing system 100 also includes a system memory 104, and a system bus 106 that couples various system components including the system memory 104 to the processing unit 102. The system bus 106 might be any of several types of bus structures including a memory bus, or memory controller; a peripheral bus; and a local bus using any of a variety of bus architectures.

Preferably, the system memory 104 includes read only memory (ROM) 108 and random access memory (RAM) 110. A basic input/output system 112 (BIOS), containing the basic routines that help transfer information between elements within the computing system 100, such as during start-up, is typically stored in the ROM 108.

Preferably, the computing system 100 further includes a secondary storage device 113, such as a hard disk drive, for reading from and writing to a hard disk (not shown), a magnetic disk drive 114 for reading from or writing to a removable magnetic disk 116, and an optical disk drive 118 for reading from or writing to a removable optical disk 119, such as a compact disk (CD) ROM, digital video disk (DVD) ROM, or other optical media.

The hard disk drive 113, magnetic disk drive 114, and optical disk drive 118 are connected to the system bus 106 by a hard disk drive interface 120, a magnetic disk drive interface 122, and an optical drive interface 124, respectively. The drives and their associated computer-readable media provide nonvolatile storage of computer readable instructions, data structures, program modules and other data for the computing system 100.

Although the exemplary environment described herein employs a hard disk, a removable magnetic disk 116, and a removable optical disk 119, it should be appreciated by those skilled in the art that other types of computer-readable media, capable of storing data, can be used in the exemplary system. Examples of these other types of computer-readable mediums include magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, CD ROMs, DVD ROMs, random access memories (RAMs), read only memories (ROMs), and the like..

A number of program modules may be stored on the hard disk, magnetic disk 116, removable optical disk 119, ROM 108, or RAM 110, including an operating system 126, one or more application programs 128, other program modules 130, and program data 132. A user may enter commands and information into the computing

system 100 through input devices, such as a keyboard 134 and a pointing device 136, such as a mouse. Examples of other input devices might include a microphone, joystick, game pad, satellite dish, scanner, and a telephone. These and other input devices are often connected to the processing unit 102 through a serial port interface 140 that is coupled to the system bus 106. Nevertheless, these input devices also might be connected by other interfaces, such as a parallel port, game port, or a universal serial bus (USB). A display device 142, such as a monitor, is also connected to the system bus 106 via an interface, such as a video adapter 144. In addition to the display device 142, computing systems, in general, typically include other peripheral devices (not shown), such as speakers, printers, and palm devices.

The computing system 100 might also include a camera 146, such as a digital/electronic still or video camera, or film/photographic scanner, capable of capturing a sequence of images. The images are input into the computing system 100 via an appropriate camera interface 150. This camera interface 150 is connected to the system bus 106, thereby allowing the images to be routed to and stored in the RAM 110, or one of the other data storage devices associated with the computing system 100. However, it is noted that image data can be input into the computing system 100 from any of the aforementioned computer-readable media as well, without requiring the use of the camera 146.

When used in a LAN networking environment 206, Fig. 1, the computing system 100 of Fig. 2 is connected to the local network 206 through a network interface or adapter 152. When used in a WAN networking environment 208, such as the Internet, the computing system 100 typically includes a modem 154 or other

means, such as a direct connection, for establishing communications over the wide area network 208. The modem 154, which can be internal or external, is connected to the system bus 106 via the serial port interface 140. In a networked environment, program modules depicted relative to the computing system 100, or portions thereof, may be stored in a remote memory storage device. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computing systems may be used.

Of course in alternate embodiments, a computing device may include any suitable computing architecture, including hand-held computers, palm computers, and small devices, such as cellular telephones.

Referring now to Fig. 3, an offloading system 300 for offloading compilation from a first subsystem to a second subsystem is shown. A receive module 301 in the first subsystem receives a call for a program A machine-readable code from the second subsystem. A retrieve operation 303 retrieves program A information, i.e. compilation information from the second subsystem. A transmit module 305 transmits the program A information from the first subsystem to the second subsystem.

A receipt module 307 receives the program A information in the second subsystem. A decode operation 309 decodes the program A information. A retrieve step 310 retrieves program A code for compilation into program A machine-readable code. It is noted that prior to compilation, the program A code might be in IL code or source code and might reside in the first subsystem, the second subsystem, or a third subsystem. Operation 311 in the second subsystem compiles the program A into machine-readable code based on the program A information received from the first

subsystem. A send module 313 sends the program A machine-readable code to the first subsystem. A receive step 315 receives the program A machine-readable code. A use module 317 uses the program A machine-readable code by, for example, storing the program A machine-readable code or executing the program A machine-readable code.

In an alternative embodiment, a first subsystem might send a compiled program to the second subsystem without a request form the second subsystem, for the second subsystem's use. In another alternative embodiment, a first subsystem might include a program to be compiled and sends both the compilation information and the program to be compiled to a second computing subsystem. The second computing subsystem then returns the compiled program to the first subsystem.

Fig. 4 is a flow chart representing logical operations of an offloading system 400 for offloading compilation of code into machine-executable code, according to another embodiment. Entrance to the operational flow of the acceptance system 400 begins at a flow connection 402. A request module in 404 subsystem receives a request for compilation on a first subsystem. The request may be generated by the first subsystem in response to a prompting, for example a request to execute a program on the first subsystem. A transmit operation 406 transmits compilation information from the first subsystem to a second subsystem. The compilation information contains information about the first subsystem and its requirements for the format of the machine-executable code. This information might also include the size of the memory or secondary storage space of the first subsystem. Preferably, the compilation code is written in IL code. In an alternative embodiment, the compilation information might be

written in source code. In another alternative embodiment, the compilation information might be data.

A get operation 408 obtains the computer program code that needs compilation by the second subsystem. Preferably, the get operation 408 obtains IL code for compilation into machine-readable code. It is noted that the get operation 408 could obtain source code for compilation into machine-readable code or into IL code, depending on the application.

In one example embodiment, the computer program code is transmitted by the first subsystem to the second subsystem along with the compilation information. In a second example embodiment, the computer program code is received by the second subsystem from a third source. In a third example embodiment, the computer program code is contained within the second subsystem for retrieval by the second subsystem.

A compile module 410 compiles the computer program code received from the get operation 408 into machine executable code for use on the first subsystem. Preferably, the compile module 410 decodes information contained within the compilation information to compile the code for the first subsystem's use. In one example embodiment, the second subsystem compiles code for multiple other subsystems each having different machine-executable code requirements. By using the compilation information received from the first subsystem, the compile module 410 can specifically compile the code needed by the first subsystem.

A transmit module 412 transmits the machine-executable code to the first subsystem. A detect module 414 identifies the source of the machine-executable code being transmitted to the first subsystem. For example, the machine-executable code

might include a data stream identifying the source. The data stream might include a digital signature for identification purposes. In a second example, the detect module 414 detects the source based on a secure link. In a third example, the detect module 414 detects the source based on a network ID or address for the second subsystem.

A query operation 416 tests whether the source is a trusted source. The query operation 416 uses a receipt policy to determine whether the source is trusted. If the query operation 416 detects that the source is a trusted source, the operational flow branches "YES" to an accept module 418. The accept module 418 accepts receipt of the machine-executable code on the first subsystem. A use step 420 uses the machine-executable code. For example, the use step 420 can store the machine-executable code in the secondary storage system of the first subsystem. In a second example, the use step 420 can execute the machine-executable code in the first subsystem. The operational flow ends at 422.

It is noted that the detect module 414 and the query operation 416 are not necessary components of the offloading system 400. If the only time that the first subsystem is connected to the second subsystem is through a secure link, the detect module 414 and the query operation 416 are not necessary; security is provided by the secure link.

Referring back to the query operation 416, if the query operation 416 detects the source is not a trusted source, operational flow branches "NO" to a reject module 424. The reject module 424 rejects receipt of the machine-executable code. The operational flow ends at 422.

The operational flow chart depicted in Fig. 4 may best be understood in terms of application examples. In a first application example, referring to Figs. 1 and 4, the request module 404 receives a request for compilation from the fourth subsystem 204. For example, it is desired to update the operating software of a cellular phone. The transmit operation 406 transmits compilation information associated with the fourth subsystem to the first subsystem 201. The get operation 408 obtains the computer program code desired to be executed on the fourth subsystem 204. In this application example, the get operation 408 finds the computer program code, in the form of IL code, on the second subsystem 202. The computer program code is transferred from the second subsystem 202 to the first subsystem 201. In an alternative example embodiment, the computer program code is contained within the first subsystem 201 for compilation. In yet another example embodiment, the computer program code is transferred from the fourth subsystem 204 to the first subsystem 201.

The compile module 410 uses the second compiler 261 of the first computing subsystem 201 to compile the computer program code into machine-executable code, specifically formatted for the fourth subsystem 204 based on the compilation information received from the fourth subsystem 204. The transmit module 412 transmits the machine-executable code from the first subsystem 201 to the fourth subsystem 204.

The detect module 414 detects that the machine-executable code is being received from the first subsystem 201. The query operation 416 detects that the first subsystem 201 is a trusted source using the fourth receipt policy 244. For example, the fourth receipt policy 244 might indicate that any source connected to the fourth

computing subsystem 204 via a dedicated line is a trusted source. In another alternative embodiment, the fourth receipt policy 244 might indicate that a source, which has a coded signature or certificate that matches a predetermined criteria, is a trusted source. In other embodiments, any suitable method for indicating a trusted source might be used. All other sources are not trusted sources. The operational flow branches "YES" to the accept module 418. The accept module 418 accepts receipt of the machine-executable code from the first subsystem 201.

The use module 420 stores the machine-executable code in the secondary storage of the fourth subsystem 204 for later execution by the fourth subsystem 204. The operational flow ends at 422.

In a second application example, the query operation 416 detects that the source of the machine-executable code is not from a trusted source, using the above stated fourth receipt policy 244. For example, the third computing subsystem 203 transmits machine-executable code to the fourth subsystem 204 via a wireless connection. The fourth receipt policy 244 indicates a wireless connection is not a trusted source. The operational flow branches "NO" to the reject module 424. The reject module 424 rejects receipt of the machine-executable code. The operational flow ends at 422.

The logical operations of the various embodiments illustrated herein are implemented (1) as a sequence of computer implemented steps running on a computing system and/or (2) as interconnected machine modules within the computing system. The implementation is a matter of choice dependent on the performance requirements of the computing system implementing the invention. Accordingly, the logical operations

making up the embodiments of the present invention described herein are referred to variously as operations, steps, engines, or modules.

The various embodiments described above are provided by way of illustration only and should not be construed to limit the invention. Those skilled in the art will readily recognize various modifications and changes that may be made to the present invention without following the example embodiments and applications illustrated and described herein, and without departing from the true spirit and scope of the present invention, which is set forth in the following claims.

CLAIMS

1. A method of offloading compilation, the method comprising:

transmitting compilation information from a first subsystem to a second subsystem;

compiling computer program code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem; and

receiving the machine-executable code from the second subsystem into the first subsystem.

2. A method according to claim 1, wherein the step of transmitting compilation information includes transmitting compilation information from a first subsystem to a second subsystem in response to a request to compile computer program code into machine-executable code.

3. A method according to claim 1, wherein the step of transmitting compilation information includes transmitting compilation information written in intermediate language code from a first subsystem to a second subsystem.

4. A method according to claim 1, wherein the step of transmitting compilation information includes transmitting compilation information from a small device to a second subsystem.

5. A method according to claim 4, wherein the step of transmitting compilation information includes transmitting compilation information from a cellular phone to a second subsystem.

6. A method according to claim 1, wherein the step of compiling computer program code includes compiling intermediate language code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem.

7. A method according to claim 1, further comprising:

before receiving the machine executable code, detecting whether the second subsystem is a trusted source.

8. A method according to claim 7, wherein the step of detecting includes using a receipt policy to detect whether the second subsystem is a trusted source.

9. A method according to claim 8, wherein the step of detecting includes detecting whether the first and second subsystem are connected via a secure link.

10. A method according to claim 1, further comprising:

using the machine-executable code on the first subsystem.

11. A method according to claim 10, wherein the step of using includes storing the machine-executable code on the first subsystem.

12. A method according to claim 11, wherein the step of using includes executing the machine-executable code on the first subsystem.

13. A method according to claim 1, wherein the step of transmitting includes transmitting compilation information and computer program code from a first subsystem to a second subsystem.

14. A method according to claim 1, further comprising:

before the step of compiling, retrieving computer program code for compilation into machine-executable code.

15. A method according to claim 14, wherein the step of retrieving computer program code includes retrieving computer program code from a third subsystem.

16. A method according to claim 1, wherein the step of compiling includes decoding the compilation information.

17. A method according to claim 1, wherein the step of transmitting compilation information from a first subsystem to a second subsystem includes transmitting compilation information from a first subsystem to a second subsystem wherein the first and second subsystems are components of a single system.

18. A method according to claim 1, wherein the step of transmitting includes transmitting compilation instructions from a first subsystem to a second subsystem.

19. A method in a first subsystem for compiling program code for execution in a second subsystem, the method comprising:

receiving compilation information from the second subsystem;

compiling computer program code into machine-executable code based on the compilation information received from the second subsystem; and

transmitting the machine-executable code to the second subsystem.

20. A method according to claim 19, wherein the step of receiving compilation information includes receiving compilation information from a small device.

21. A method according to claim 19, wherein the step of receiving compilation information includes receiving compilation information from a cellular phone.

22. A method according to claim 19, wherein the step of compiling computer program code includes compiling intermediate language code into machine-executable code based on the compilation information received from the second subsystem.

23. A method according to claim 19, wherein the step of receiving includes receiving compilation information and computer program code from the second subsystem.

24. A method according to claim 19, further comprising:

before the step of compiling, retrieving computer program code for compilation into machine-executable code.

25. A method according to claim 24, wherein the step of retrieving computer program code includes retrieving computer program code from the first subsystem.

26. A method according to claim 19, wherein the step of compiling includes decoding the compilation information.

27. A method in a second subsystem for offloading compilation to a first subsystem having a program code compiler, the method comprising:

transmitting compilation information to the first subsystem; and

receiving machine-executable code, compiled from the compilation information, from the first subsystem.

28. A method according to claim 27, wherein the step of transmitting compilation information includes transmitting compilation information in response to a request to compile computer program code into machine-executable code.

29. A method according to claim 27, wherein the step of transmitting compilation information includes transmitting compilation information written in intermediate language code.

30. A method according to claim 27, further comprising:

before receiving machine executable code, detecting whether the first subsystem is a trusted source.

31. A method according to claim 30, wherein the step of detecting includes using a receipt policy to detect whether the first subsystem is a trusted source.

32. A method according to claim 31, wherein the step of detecting includes detecting whether the first subsystem is connected via a secure link.

33. A method according to claim 27, wherein the step of transmitting includes transmitting compilation information and computer program code to a first subsystem.

34. A computer program storage medium readable by a computing system and encoding a computer program of instructions for executing a computer process for offloading compilation, the computer process comprising:

sending program information from a first subsystem to a second subsystem;

compiling program code into machine-executable code on the second subsystem based on the program information received from the first subsystem; and

receiving the machine-executable code from the second subsystem into the first subsystem.

35. A computer process according to claim 34, wherein the step of sending program information includes sending program information from a first subsystem to a second subsystem in response to a request to compile program code into machine-executable code.

36. A computer process according to claim 34, wherein the step of sending program information includes sending program information written in intermediate language code from a first subsystem to a second subsystem.

37. A computer process according to claim 34, wherein the step of sending program information includes sending program information from a small device to a second subsystem.

38. A computer process according to claim 37, wherein the step of sending program information includes sending program information from a cellular phone to a second subsystem.

39. A computer process according to claim 34, wherein the step of compiling program code includes compiling intermediate language code into machine-executable code on the second subsystem based on the program information received from the first subsystem.

40. A computer process according to claim 34, further comprising:

before receiving the machine executable code, detecting whether the second subsystem is a trusted source.

41. A computer process according to claim 40, wherein the step of detecting includes using a receipt policy to detect whether the second subsystem is a trusted source.

42. A computer process according to claim 41, wherein the step of detecting includes detecting whether the first and second subsystem are connected via a secure link.

43. A computer process according to claim 34, wherein the step of sending includes sending program information and computer program code from a first subsystem to a second subsystem.

44. A computer process according to claim 34, further comprising:

before the step of compiling, retrieving program code for compilation into machine-executable code.

45. A computer process according to claim 44, wherein the step of retrieving program code includes retrieving program code from a third subsystem.

46. A computer process according to claim 34, wherein the step of compiling includes decoding the program information.

47. A computer process according to claim 34, wherein the step of sending includes sending compilation instructions from a first subsystem to a second subsystem.

48. A system for offloading compilation, the apparatus comprising:

a transmit module that transmits compilation information from a first subsystem to a second subsystem;

a compile module that compiles program code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem; and

a receive module that receives the machine-executable code from the second subsystem into the first subsystem.

49. A system according to claim 48, wherein the transmit module transmits in response to a request to compile program code into machine-executable code.

50. A system according to claim 48, wherein the compilation information is written in intermediate language code.

51. A system according to claim 48, wherein the first subsystem is a small device.

52. A system according to claim 51, wherein the small device is a cellular phone.

53. A system according to claim 48, further comprising:

a detect module that detects whether the second subsystem is a trusted source.

54. A system according to claim 53, wherein the detect module uses a receipt policy to detect whether the second subsystem is a trusted source.

55. A system according to claim 54, wherein the detect module detects whether the first and second subsystem are connected via a secure link.

56. A computer data signal embodied in a carrier wave readable by a computing system and encoding a computer program of instructions for executing a computer process for offloading compilation, the computer process comprising:

sending program information from a first subsystem to a second subsystem;

compiling program code into machine-executable code on the second subsystem based on the program information received from the first subsystem; and

receiving the machine-executable code from the second subsystem into the first subsystem.

57. A computer process according to claim 56, wherein the step of sending program information includes sending program information from a first subsystem to a second subsystem in response to a request to compile program code into machine-executable code.

58. A computer process according to claim 57, wherein the step of sending program information includes sending program information written in intermediate language code from a first subsystem to a second subsystem.

59. A computer process according to claim 57, wherein the step of sending program information includes sending program information from a small device to a second subsystem.

60. A computer process according to claim 59, wherein the step of sending program information includes sending program information from a cellular phone to a second subsystem.

61. A computer process according to claim 56, wherein the step of compiling program code includes compiling intermediate language code into machine-executable code on the second subsystem based on the program information received from the first subsystem.

62. A computer process according to claim 56, further comprising:

before receiving the machine executable code, detecting whether the second subsystem is a trusted source.

63. A computer process according to claim 62, wherein the step of detecting includes using a receipt policy to detect whether the second subsystem is a trusted source.

64. A computer process according to claim 63, wherein the step of detecting includes detecting whether the first and second subsystem are connected via a secure link.

65. A computer process according to claim 56, wherein the step of sending includes sending program information and computer program code from a first subsystem to a second subsystem.

66. A computer process according to claim 56, further comprising:

before the step of compiling, retrieving program code for compilation into machine-executable code.

67. A computer process according to claim 66, wherein the step of retrieving program code includes retrieving program code from a third subsystem.

68. A computer process d according to claim 56, wherein the step of compiling includes decoding the program information.

ABSTRACT OF THE INVENTION

A method of offloading compilation includes transmitting compilation information from a first subsystem to a second subsystem. The method also includes compiling computer program code into machine-executable code on the second subsystem based on the compilation information received from the first subsystem. The method further includes receiving the machine-executable code from the second subsystem into the first subsystem.

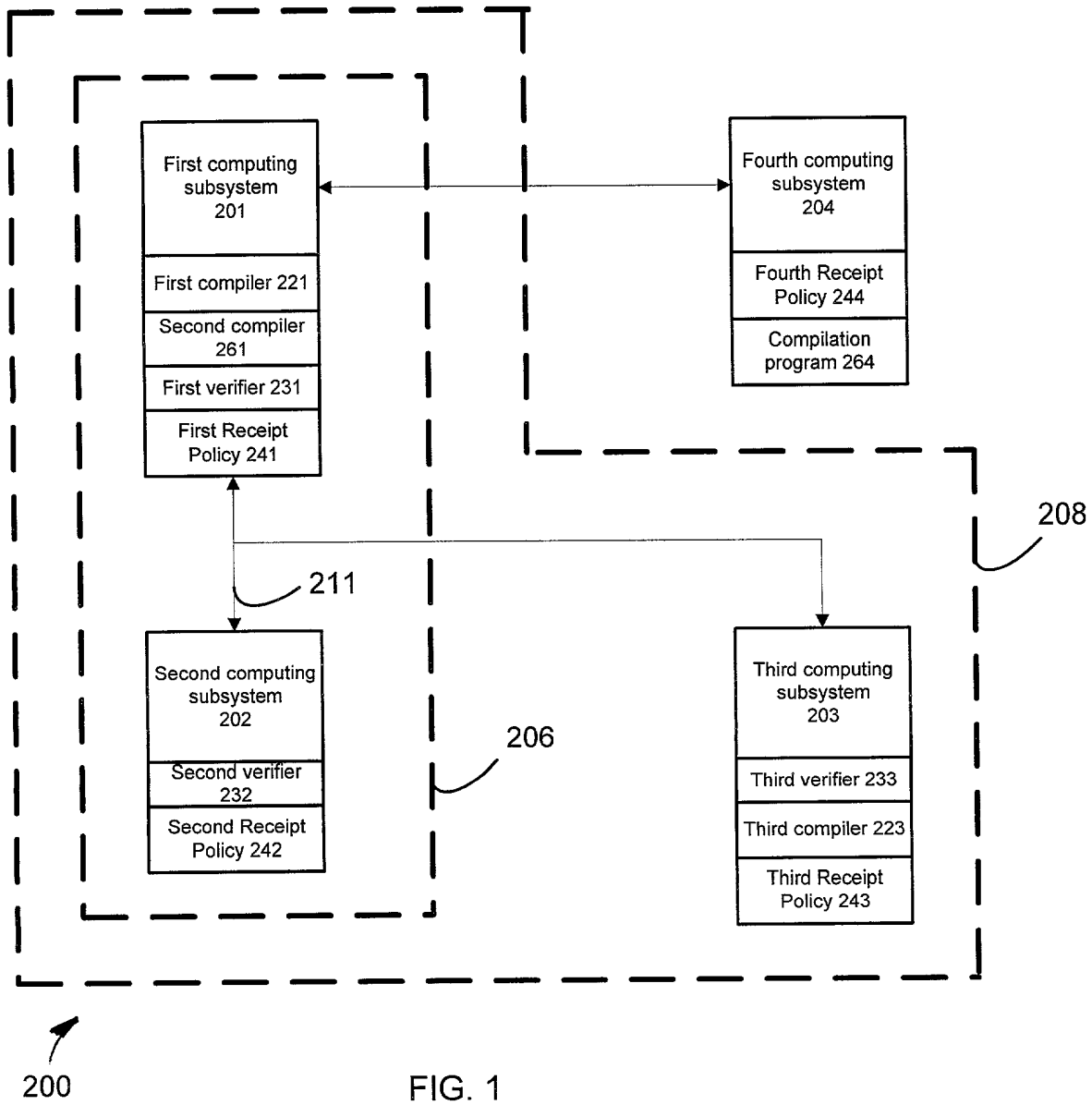
First computing subsystem 201
First compiler 221
Second compiler 261
First verifier 231
First Receipt Policy 241
Fourth computing subsystem 204
Fourth Receipt Policy 244
Compilation program 264
208
211
Second computing subsystem 202
Second verifier 232
Second Receipt Policy 242
206
Third computing subsystem 203
Third verifier 233
Third compiler 223
Third Receipt Policy 243
200

FIG. 1

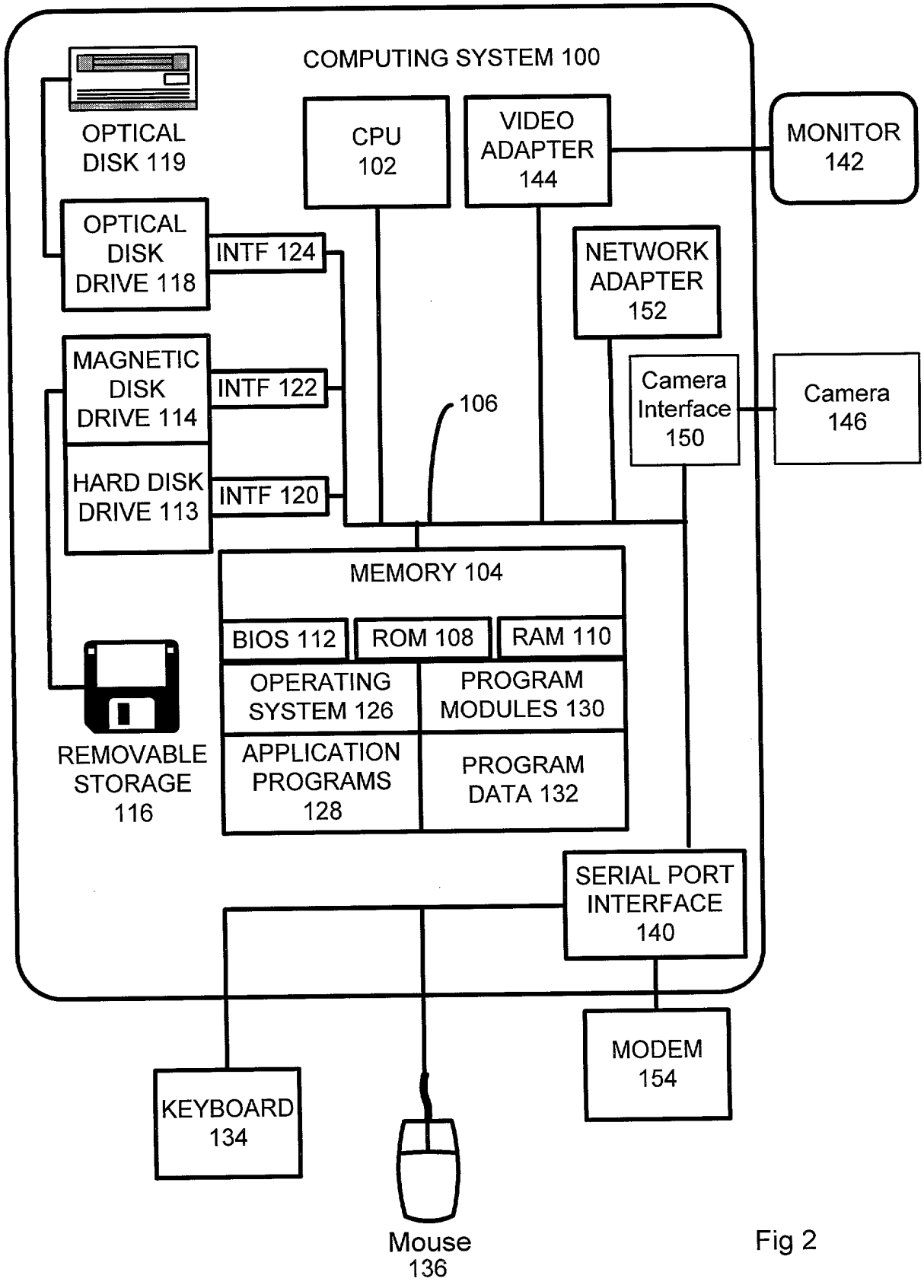
COMPUTING SYSTEM 100
OPTICAL DISK 119
CPU 102
VIDEO ADAPTER 144
MONITOR 142
OPTICAL DISK DRIVE 118
INTF 124
NETWORK ADAPTER 152
MAGNETIC DISK DRIVE 114
INTF 122
106
Camera Interface 150
Camera 146
HARD DISK DRIVE 113
INTF 120
MEMORY 104
BIOS 112
ROM 108
RAM 110
OPERATING SYSTEM 126
PROGRAM MODULES 130
APPLICATION PROGRAMS 128
PROGRAM DATA 132
REMOVABLE STORAGE 116
SERIAL PORT INTERFACE 140
MODEM 154
KEYBOARD 134
Mouse 136

Fig 2

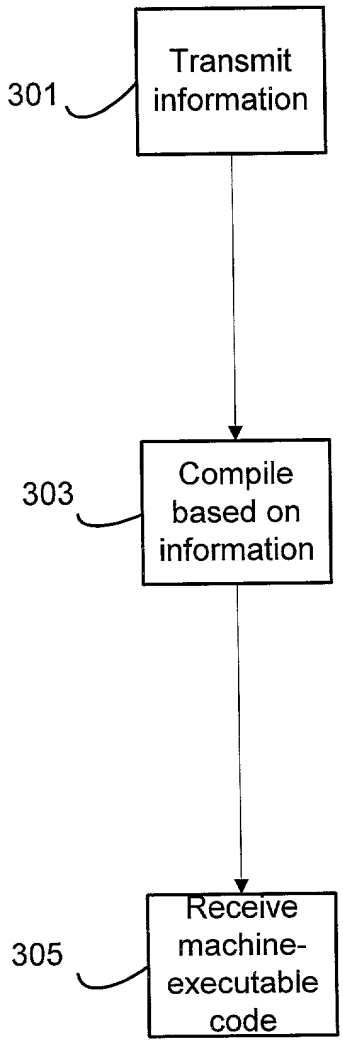
Transmit information
301
Compile based on information
303
Receive machine-executable code
305
300

FIG. 3

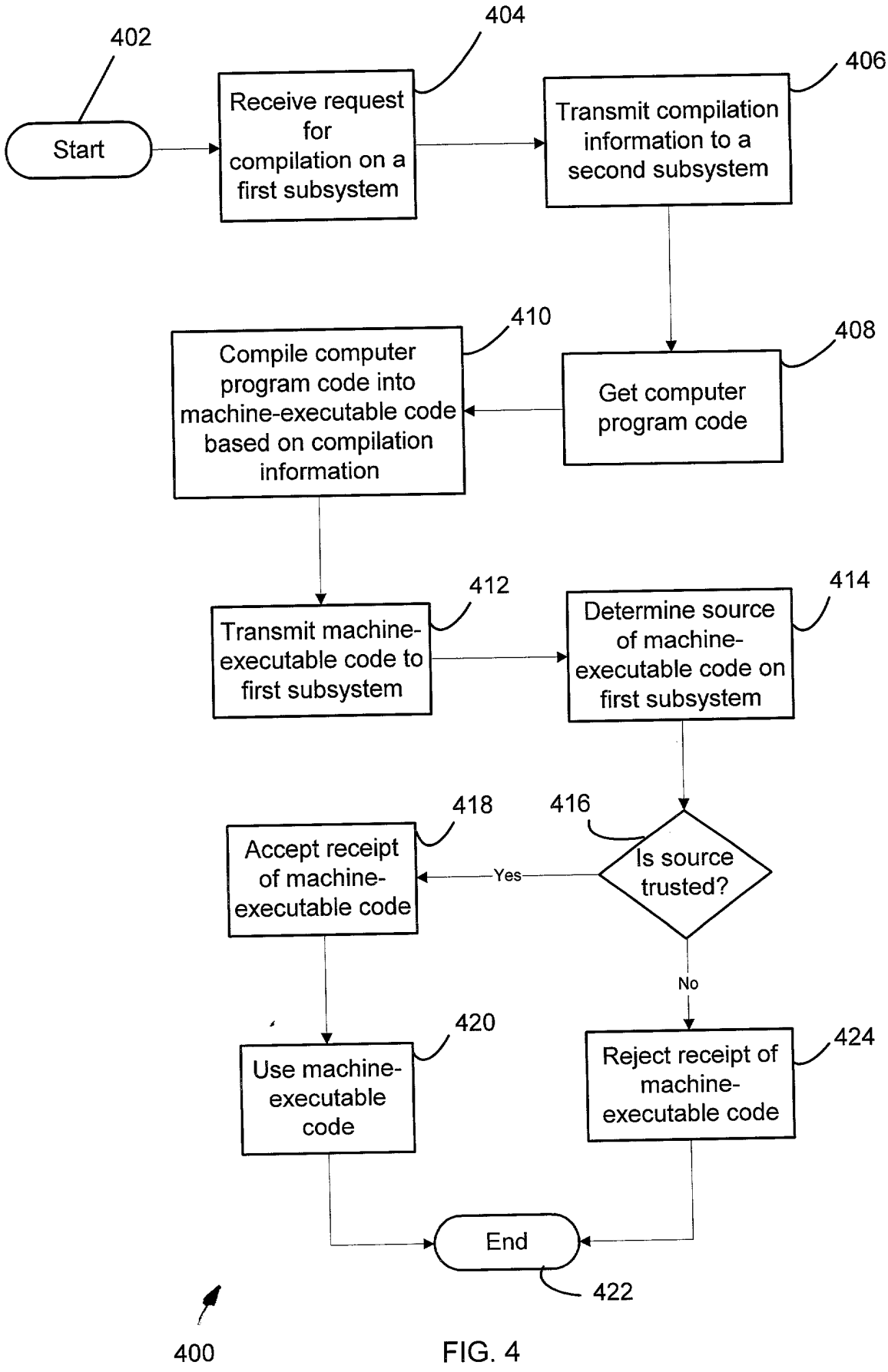
402
Start
404
Receive request for compilation on a first subsystem
406
Transmit compilation information to a second subsystem
408
Get computer program code
410
Compile computer program code into machine-executable code based on compilation information
412
Transmit machine-executable code to first subsystem
414
Determine source of machine-executable code on first subsystem
416
Is source trusted?
Yes
No
418
Accept receipt of machine-executable code
420
Use machine-executable code
424
Reject receipt of machine-executable code
End
422
400

FIG. 4

MERCHANT & GOULD P.C.

United States Patent Application

COMBINED DECLARATION AND POWER OF ATTORNEY

As a below named inventor I hereby declare that my residence, post office address and citizenship are as stated below next to my name; that

I verily believe I am the original, first and a joint inventor of the subject matter which is claimed and for which a patent is sought on the invention entitled METHOD AND SYSTEM FOR ACCEPTING PRECOMPILED INFORMATION, the specification of which I have reviewed and for which I solicit a United States patent hereto **OR** [filed on _____ and identified in the U.S. Patent and Trademark Office as Serial No. _____.]

I hereby state that I have reviewed and understand the contents of the above–identified specification, including the claims.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, § 1.56 (attached hereto).

I hereby claim foreign priority benefits under Title 35, United States Code, § 119/365 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on the basis of which priority is claimed:

a. ☒ no such applications have been filed.
b. ☐ such applications have been filed as follows:

FOREIGN APPLICATION(S), IF ANY, CLAIMING PRIORITY UNDER 35 USC § 119			
COUNTRY	**APPLICATION NUMBER**	**DATE OF FILING (day, month, year)**	**DATE OF ISSUE (day, month, year)**
ALL FOREIGN APPLICATION(S), IF ANY, FILED BEFORE THE PRIORITY APPLICATION(S)			
COUNTRY	**APPLICATION NUMBER**	**DATE OF FILING (day, month, year)**	**DATE OF ISSUE (day, month, year)**

I hereby claim the benefit under Title 35, United States Code, § 120/365 of any United States and PCT international application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, § 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, § 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application.

U.S. APPLICATION NUMBER	DATE OF FILING (day, month, year)	STATUS (patented, pending, abandoned)

I hereby claim the benefit under Title 35, United States Code § 119(e) of any United States provisional application(s) listed below:

U.S. PROVISIONAL APPLICATION NUMBER	DATE OF FILING (Day, Month, Year)

I hereby appoint the following attorney(s) and/or patent agent(s) to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith:

Albrecht, John W.	Reg. No. 40,481	Lacy, Paul E.	Reg. No. 38,946
Anderson, Gregg I.	Reg. No. 28,828	Larson, James A.	Reg. No. 40,443
Ansems, Gregory M.	Reg. No. 42,264	Liepa, Mara E.	Reg. No. 40,066
Batzli, Brian H.	Reg. No. 32,960	Lindquist, Timothy A.	Reg. No. 40,701
Beard, John L.	Reg. No. 27,612	Lycke, Lawrence E.	Reg. No. 38,540
Berns, John M.	Reg. No. 43,496	McAuley, Steven A.	Reg. No. P–46,084
Black, Bruce E.	Reg. No. 41,622	McDonald, Daniel W.	Reg. No. 32,044
Branch, John W.	Reg. No. 41,633	McIntyre, Jr., William F.	Reg. No. P–44,921
Bremer, Dennis C.	Reg. No. 40,528	Mueller, Douglas P.	Reg. No. 30,300
Bruess, Steven C.	Reg. No. 34,130	Pauly, Daniel M.	Reg. No. 40,123
Byrne, Linda M.	Reg. No. 32,404	Phillips, John B.	Reg. No. 37,206
Carlson, Alan G.	Reg. No. 25,959	Plunkett, Theodore	Reg. No. 37,209
Caspers, Philip P.	Reg. No. 33,227	Prendergast, Paul J.	Reg. No. 46,068
Chiapetta, James R.	Reg. No. 39,634	Pytel, Melissa J.	Reg. No. 41,512
Clifford, John A.	Reg. No. 30,247	Qualey, Terry	Reg. No. 25,148
Cochran, William W.	Reg. No. 26,652	Reich, John C.	Reg. No. 37,703
Daignault, Ronald A.	Reg. No. 25,968	Reiland, Earl D.	Reg. No. 25,767
Daley, Dennis R.	Reg. No. 34,994	Schmaltz, David G.	Reg. No. 39,828
Dalglish, Leslie E.	Reg. No. 40,579	Schuman, Mark D.	Reg. No. 31,197
Daulton, Julie R.	Reg. No. 36,414	Schumann, Michael D.	Reg. No. 30,422
Devries Smith, Katherine M.	Reg. No. 42,157	Scull, Timothy B.	Reg. No. 42,137
DiPietro, Mark J.	Reg. No. 28,707	Sebald, Gregory A.	Reg. No. 33,280
Edell, Robert T.	Reg. No. 20,187	Skoog, Mark T.	Reg. No. 40,178
Epp Ryan, Sandra	Reg. No. 39,667	Spellman, Steven J.	Reg. No. 45,124
Glance, Robert J.	Reg. No. 40,620	Stoll–DeBell, Kirstin L.	Reg. No. 43,164
Goggin, Matthew J.	Reg. No. 44,125	Storer, Shelley D.	Reg. No. P45,135
Golla, Charles E.	Reg. No. 26,896	Sumner, John P.	Reg. No. 29,114
Gorman, Alan G.	Reg. No. 38,472	Sumners, John S.	Reg. No. 24,216
Gould, John D.	Reg. No. 18,223	Swenson, Erik G.	Reg. No. 45,147
Gregson, Richard	Reg. No. 41,804	Tellekson, David K.	Reg. No. 32,314
Gresens, John J.	Reg. No. 33,112	Trembath, Jon R.	Reg. No. 38,344
Hamre, Curtis B.	Reg. No. 29,165	Underhill, Albert L.	Reg. No. 27,403
Hillson, Randall A.	Reg. No. 31,838	Vandenburgh, J. Derek	Reg. No. 32,179
Holzer, Jr., Richard J.	Reg. No. 42,668	Wahl, John R.	Reg. No. 33,044
Johnston, Scott W.	Reg. No. 39,721	Weaver, Karrie G.	Reg. No. 43,245
Kadievitch, Natalie D.	Reg. No. 34,196	Welter, Paul A.	Reg. No. 20,890
Karjeker, Shaukat	Reg. No. 34,049	Whipps, Brian	Reg. No. 43,261
Kastelic, Joseph M.	Reg. No. 37,160	Wickhem, J. Scot	Reg. No. 41,376
Kettelberger, Denise	Reg. No. 33,924	Williams, Douglas J.	Reg. No. 27,054
Keys, Jeramie J.	Reg. No. 42,724	Witt, Jonelle	Reg. No. 41,980
Knearl, Homer L.	Reg. No. 21,197	Wu, Tong	Reg. No. 43,361
Kowalchyk, Alan W.	Reg. No. 31,535	Xu, Min S.	Reg. No. 39,536
Kowalchyk, Katherine M.	Reg. No. 36,848	Zeuli, Anthony R.	Reg. No. P45,255

In addition, I also hereby appoint the following attorneys to prosecute this application and to transact all business in the U.S. Patent and Trademark Office in connection therewith:

Kate E. Sako, Reg, No. 32,628
Daniel D. Crouse, Reg. No. 32,022

I hereby authorize them to act and rely on instructions from and communicate directly with the person/assignee/attorney/firm/ organization who/which first sends/sent this case to them and by whom/which I hereby declare that I have consented after full disclosure to be represented unless/until I instruct Merchant & Gould P.C. to the contrary.

Please direct all correspondence in this case to Merchant & Gould P.C. at the address indicated below:

Homer L. Knearl
Merchant & Gould P.C.
3100 Norwest Center
90 South Seventh Street
Minneapolis, MN 55402-4131
303.357.1633

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

2	**Full Name Of Inventor**	**Family Name** SMITH	**First Given Name** MICHAEL	**Second Given Name** D.
0	**Residence & Citizenship**	**City** KIRKLAND	**State or Foreign Country** WASHINGTON	**Country of Citizenship** UNITED KINGDOM
1	**Post Office Address**	**Post Office Address** 1814 FIRST STREET	**City** KIRKLAND	**State & Zip Code/Country** WASHINGTON/USA
Signature of Inventor 201:			**Date:**	

2	**Full Name Of Inventor**	**Family Name** MOMOH	**First Given Name** OSHOMA	**Second Given Name**
0	**Residence & Citizenship**	**City** SEATTLE	**State or Foreign Country** WASHINGTON	**Country of Citizenship** USA
2	**Post Office Address**	**Post Office Address** 400 HARVARD AVENUE EAST, #302	**City** SEATTLE	**State & Zip Code/Country** WASHINGTON 98102/USA
Signature of Inventor 202:			**Date:**	

§ 1.56 Duty to disclose information material to patentability.

(a) A patent by its very nature is affected with a public interest. The public interest is best served, and the most effective patent examination occurs when, at the time an application is being examined, the Office is aware of and evaluates the teachings of all information material to patentability. Each individual associated with the filing and prosecution of a patent application has a duty of candor and good faith in dealing with the Office, which includes a duty to disclose to the Office all information known to that individual to be material to patentability as defined in this section. The duty to disclose information exists with respect to each pending claim until the claim is canceled or withdrawn from consideration, or the application becomes abandoned. Information material to the patentability of a claim that is canceled or withdrawn from consideration need not be submitted if the information is not material to the patentability of any claim remaining under consideration in the application. There is no duty to submit information which is not material to the patentability of any existing claim. The duty to disclose all information known to be material to patentability is deemed to be satisfied if all information known to be material to patentability of any claim issued in a patent was cited by the Office or submitted to the Office in the manner prescribed by §§ 1.97(b)-(d) and 1.98. However, no patent will be granted on an application in connection with which fraud on the Office was practiced or attempted or the duty of disclosure was violated through bad faith or intentional misconduct. The Office encourages applicants to carefully examine:

(1) prior art cited in search reports of a foreign patent office in a counterpart application, and

(2) the closest information over which individuals associated with the filing or prosecution of a patent application believe any pending claim patentably defines, to make sure that any material information contained therein is disclosed to the Office.

(b) Under this section, information is material to patentability when it is not cumulative to information already of record or being made of record in the application, and

(1) It establishes, by itself or in combination with other information, a prima facie case of unpatentability of a claim;

or

(2) It refutes, or is inconsistent with, a position the applicant takes in:

(i) Opposing an argument of unpatentability relied on by the Office, or

(ii) Asserting an argument of patentability.

A prima facie case of unpatentability is established when the information compels a conclusion that a claim is unpatentable under the preponderance of evidence, burden–of–proof standard, giving each term in the claim its broadest reasonable construction consistent with the specification, and before any consideration is given to evidence which may be submitted in an attempt to establish a contrary conclusion of patentability.

(c) Individuals associated with the filing or prosecution of a patent application within the meaning of this section are:

(1) Each inventor named in the application:

(2) Each attorney or agent who prepares or prosecutes the application; and

(3) Every other person who is substantively involved in the preparation or prosecution of the application and who is associated with the inventor, with the assignee or with anyone to whom there is an obligation to assign the application.

(d) Individuals other than the attorney, agent or inventor may comply with this section by disclosing information to the attorney, agent, or inventor.